DOCUMENT RESUME

ED 464 608 IR 021 191

AUTHOR Foshay, Rob; Ahmed, Muhammad I.
TITLE A Practical Process for Reviewing and Selecting Educational Software. Technical Paper.
INSTITUTION PLATO Learning, Inc., Bloomington, MN.
REPORT NO PLATO-TP-8
PUB DATE 2000-09-00
NOTE 41p.
AVAILABLE FROM For full text: http://www.plato.com/whitepapers.asp.
PUB TYPE Guides - Non-Classroom (055) -- Tests/Questionnaires (160)
EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS Computer Software; *Computer Software Selection; Computer Uses in Education; *Courseware; Educational Media; *Educational Technology; *Evaluation Criteria; Guidelines; Instructional Effectiveness; Instructional Materials; *Media Selection; *Selection Tools

ABSTRACT

This paper provides a balanced set of guidelines for reviewing educational software, and offers a simple and practical procedure that can be used to select software for instructional purposes. The procedure applies equally well to individual "titles" or to large-scale curriculum-based systems. It applies to software delivered on a CD-ROM, through a Local Area Network, or via the Internet (PLATO[R] provides all of these kinds of software and delivers via all these means). Using the procedure, the software's four subsystems--software, content, instructional, and instructional management and assessment--must first be examined. Then, additional quality criteria can be applied which are specific to each major type of software: drill-and-practice (and most tests); tutorial; simulation (and some assessments); reference/informational; and tool. A checklist is included which outlines a process involving five global judgments that should be made about any educational software product. (Contains 21 references.) (AEF)

Reproductions supplied by EDRS are the best that can be made
from the original document.

ED 464 608

PLATO®

A Practical Process for Reviewing and Selecting Educational Software

Technical Paper #8

September, 2000

Rob Foshay, Ph.D.
PLATO Learning, Inc.

Muhammad I. Ahmed
Indiana University

PLATO Learning, Inc.
10801 Nesbitt Avenue South
Bloomington, MN 55437

(800) 869-2000
http://www.plato.com

author's e-mail:
rfoshay@plato.com

PERMISSION TO REPRODUCE AND DISSEMINATE THIS MATERIAL HAS BEEN GRANTED BY

W.R. Foshay

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

IR021191

COPYRIGHT © PLATO LEARNING, INC., 2000. ALL RIGHTS RESERVED. MAY BE DUPLICATED AND DISTRIBUTED, WITH CREDIT TO PLATO LEARNING, INC.

BEST COPY AVAILABLE

Table of Contents

Educational Software as a Learning System ____ 3

The Five Basic Types of Educational Software ____ 5

Criteria for the Subsystems ____ *9*

Software Subsystem ____ 9

The Instructional Subsystem ____ 12

Content/Information Subsystem ____ 14

Instructional Management & Assessment Subsystem ____ 16

Additional Criteria for the Five Types of Educational Software ____ *19*

Drill-and-Practice ____ 20

Tutorials ____ 21

Instructional Simulations and Games ____ 22

Informational Software ____ 24

Tools ____ 26

Which Type of Software Should I Use? ____ 27

Software Type Ratings: A Checklist ____ 33

COPYRIGHT © PLATO LEARNING, INC., 2000. ALL RIGHTS RESERVED. MAY BE DUPLICATED AND DISTRIBUTED, WITH CREDIT TO PLATO LEARNING, INC.

Part 1

Introduction

It's hard to make the right choices when you must allocate limited funds for educational software. The number of factors to consider when selecting software is very large. The task of matching available software to the needs of your program is extremely complex.

The purpose of this paper is to provide a balanced set of guidelines for reviewing educational software, and to give you a simple and practical procedure you can use to select software for instructional purposes. The procedure applies equally well to individual "titles" or to large-scale curriculum-based systems. It applies to software delivered on a CD-ROM, through a Local Area Network, or via the Internet (PLATO provides all of these kinds of software and delivers via all these means). You can use it whether you are making long-term decisions on software adoption, or short-term decisions on what to use Monday. If your goal is educational effectiveness, this procedure will help you decide what to buy based on your needs.

Using the procedure, you first examine the software's four *subsystems*. Then you can apply additional *quality criteria, which* are specific to each major type of software. Thus, for any given software product, you can make your selections based on five simple judgements based on five short checklists.

If you want to learn about the rationale for these guidelines, keep reading the next two sections.

If you want to start using the procedure right now, skip ahead to Part 4.

Copyright 2000 by PLATO Learning, Inc.

Part 2

A Practical Perspective on Educational Software

Educational software can be reviewed from multiple perspectives. Often, producers and reviewers pay attention to either the appearance of the software (the *designer's perspective*) or to its functionality (the *teachers' perspective*)(Hakkinen, 1996). But this approach to reviewing software is inherently unbalanced. For example, the likely effectiveness of the software — how much and what a learner can be expected to learn (the *learner's perspective*) —often gets little attention. The *technical perspective* – what platform the software requires – often becomes the "tail wagging the dog" of educational usefulness. Similarly, reviewers rarely consider the functionality needed to manage the instruction and assess learners as part of a full instructional solution (the *management perspective,* which should be part of the teachers' perspective).

An alternative is to think of educational software as a system with four subsystems:

- The *Software* subsystem
- The *Content* subsystem
- The *Instructional* subsystem, and
- The *Instructional Management and Assessment* subsystem.

A balanced approach to software review should examine all four subsystems and their interactions and synergies.

Educational software also can be of five basic types: *tutorial*, *drill and practice, simulation/game*, *information,* and *management and assessment.* In addition to general criteria from each of the above subsystems, there are specific quality criteria, which apply individually to each type. A common error of reviewers is to apply criteria that are not appropriate to the type of software being reviewed.

The narrow focus of most reviews has inevitably shaped the software available in the market. Regardless of its technical sophistication, most educational software is instructionally of limited use. It is often creative and aesthetically appealing; but it aspires to teach only low-level learning outcomes, the instructional models

Copyright 2000 by PLATO Learning, Inc.

employed are weak, each piece of software behaves as if it were the only instructional event in the curriculum, and there is little or no meaningful provision for assessment. Integration of such software into an effective curriculum with effective instructional management is a major challenge, at best.

In this paper, we introduce a more balanced approach based on four subsystems of educational software. We will also discuss specific additional design criteria, which are important for each of the five basic types of educational software. We will call this combined approach the "practical perspective" for reviewing educational software. Where possible, the criteria are derived from standard texts and literature on instructional design and computer-based instruction. Where no suitable reference could be located, we have relied on the 40 years of experience of the PLATO organization.

Of course, any software review is a "snapshot," and products – particularly on the Internet – change constantly. For example, an Internet-based "education portal" site with daily classroom information feeds and activities changes on a daily or weekly basis. By contrast, a successful CD-ROM-based tutorial may change on 1- to 3-year upgrade cycle. Therefore, it's important to use a time frame for review decisions which is appropriate to the software type and its application. A multi-year review and adoption cycle, following the textbook adoption model, may not be well suited to all of your needs for software; you are more likely to require a combination of review cycle times and processes to meet your varied needs.

Now, let's look at the systems rationale behind the practical perspective for software review. Then, we'll examine the five types of software you are likely to encounter. In the Part 3, we'll look at the quality criteria for the four subsystems and five types. In Part 4, we'll provide a checklist in a simple format you can adapt for your needs.

Educational Software as a Learning System

We said above that you can view educational software as consisting of four subsystems. The subsystems are described in Table 1, below:

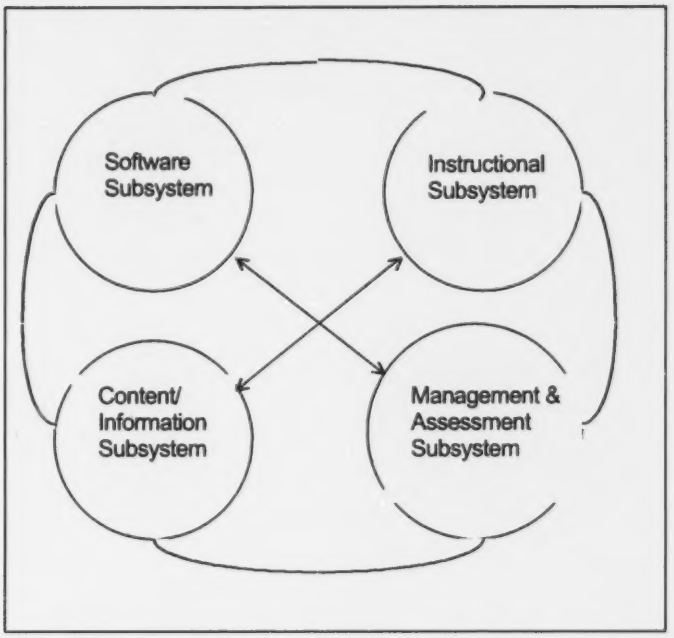
Copyright 2000 by PLATO Learning, Inc.

Software	The hardware/software environment in which the software is intended to run, and the interface with users. Includes assumptions and requirements about type of computer, display and peripherals, operating system, connectivity, overall power, performance, reliability and interoperability. Also includes assumptions about the knowledge, sophistication and goals of users.
Instruction	The way in which teachers and learners are expected to interact with the software when learning. Includes assumptions about role and characteristics of teachers and learners, and the nature of knowledge and learning.
Content/Information	The facts, concepts, principles and procedural knowledge provided for learning. Includes assumptions about the goals of teachers and learners, the structure of the knowledge and the curriculum, and the prior knowledge and abilities of the learners.
Instructional management and assessment.	The way the software handles and reports information about the learner and the instructor; assesses learning outcomes before, during and after the learning event; and makes and supports decisions about what to do next based on information available. Includes assumptions about measurement of learning, the structure of curriculum and of learning environments, the role and needs of all stakeholders in the teaching/learning process: teachers, learners, administrators, parents, and the general public.

Table 1: The Four Subsystems of Educational Software

To provide an effective learning system, the constituent subsystems must be linked to and supported by each other. If any of the subsystems is missing or is poorly implemented, the whole learning system will be greatly weakened. The concept of educational software as a learning system with its constituent subsystems is shown in Figure 1 on the next page.

Copyright 2000 by PLATO Learning, Inc.



• Figure 1. Educational Software as a Learning System: 4 Subsystems

The better the constituent subsystems are integrated and support each other, the more effective learning will be. We believe that the best way to review educational software is to examine all four subsystems. When reviewing educational software, significant weaknesses in *any* of the four subsystems can be cause to eliminate the software from further consideration.

The Five Basic Types of Educational Software

Once you have established that the four subsystems of the software are acceptably implemented, you next need to take into account the type of software you are examining. Educational software can be broadly classified in the five types listed in Table 2. Some products are relatively "pure" examples of a single type, while others combine more than one type. For example, the PLATO Web Learning Network's architecture encompasses all five software types. PLATO tutorial lessons often also include simulation or games, and usually are accompanied by practice and assessment activities. PLATO Problem Solving Activities (PSA's) include simulations and often include tools. PLATO's CyberEd science series includes information, tools, practice and assessment.

Copyright 2000 by PLATO Learning, Inc.

Assessment is not included as a separate software type, because we believe assessment is best viewed in light of the decisions it supports, as part of the instructional management subsystem. Assessments can use the same interaction types as the five basic types of software, but with minimal "real time" feedback to the learner. Many assessments use the same interaction types common in drill and practice, and should be rated using the drill and practice checklist. Some assessments use simulations, and should be rated using that checklist.

Table 2 describes the five types of educational software. Each type of software has its own unique criteria of effectiveness, which you need to examine.

Copyright 2000 by PLATO Learning, Inc.

Drill-and-practice (and most tests)	Provides repetition and reinforcement of knowledge and skills previously learned. Includes worksheets, practice tests, and many styles of games and "edutainment." Also includes most tests.
Tutorial	Provides a complete direct teaching/learning experience, centered on presentation, practice and feedback, together with introductions, summaries, and other organizational/learner control devices.
Simulation (and some assessments)	Provides opportunities to manipulate an environment (sometimes called a *microworld)* and view the consequences of the manipulation, sometimes with supplementary feedback mechanisms which make visible aspects of the simulated system not perceived in "reality." Often compresses or expands space and/or time to make such effects more visible. If the environment captures some aspects of the "real world," the result is a *simulation.* The more fantasy/unrealistic the environment, the more likely the result will be viewed as a *game*.
Reference/ Informational	Provides factual content in the form of text, pictures and/or other multimedia. Often organized for a particular type of use (such as exploration, reference or review) by a particular type of user, but makes relatively fewer assumptions about the use and the user than is true with the three above software types. Interaction typically is limited to access/exploration, and does not include practice and feedback as in the above three types of software.
Tool	Automates (typically low-level) tasks of data preparation, analysis and organization, handling and storage. Examples include numeric data (calculators, spreadsheets, graphing and analytical tools), words (word processors, search engines, most databases), multimedia (some data bases, browsers, graphic and production tools), and communication (e-mail, browsers, chat and messaging tools). Includes neither information, instruction, practice nor simulation (though tools are often bundled with these other software types). Makes some assumptions about intended use and user, but the assumptions are relatively weak compared to the other software types, so tools are the most flexible of the software types, but require the most structure to use effectively.

Table 2: Five Basic Types of Educational Software

Copyright 2000 by PLATO Learning, Inc.

In sum, by combining the four subsystems with the appropriate software type criteria, you will need to examine any particular piece of software from a total of five perspectives. This five-way rating will give you the most complete view of the probable effectiveness of the software in your application.

In Part 3, we'll examine the criteria which apply to each of the four subsystems of the learning system, and the criteria for each of the five types of software.

Copyright 2000 by PLATO Learning, Inc.

Part 3

Rating Criteria for the Subsystems and Types

Criteria for the Subsystems

We have described the complete learning system as composed of four subsystems: software, instructional, content, and assessment/management. Your review of the characteristics of each subsystem will help you decide how useful the software will be for you.

Software Subsystem

Of course, you need to be able to run the software you select, with acceptable speed and reliability, on your hardware, under your operating system. You must be able to distribute it in the way you need, whether that is by CD-ROM, Local Area Network (LAN), or via the Internet. Sloane points out "there is an extensive interdependence between hardware and software, and inadequate hardware will cause an instructional program to fail"(Sloane & et al., 1989). Therefore, it's important to assure availability of appropriate and required hardware, operating system, networking hardware and software, and peripherals attached to workstation or network to run the educational software.

If you are planning for a new installation, you should first select the educational software you want, and then select the hardware and operating system you need to run that software. If you are acquiring software for an existing installation, you will need to assess the technical requirements of the software (often called the *minimum platform specification*). Then you can either limit your selection to software which will run on your current hardware and operating system, or you can plan to upgrade your hardware and operating system software as needed. The checklist at the end of this paper is worded to assume your hardware/software platform is already in place; modify the wording if necessary.

Either way, you should determine the technical requirements of the educational software using these criteria:

- For desktop operation, the *operating system* (usually a Windows version, or MacIntosh OS at the right version number). For many types of Internet software, *browser compatibility* is important. The most common browsers are Microsoft's Internet Explorer, and Netscape. Be sure to check version numbers as well as types of systems. Also, check if the software makes additions to the browser (called "plug-ins" and "cookies"). Some Internet security systems have policies which prevent or limit such additions.

Copyright 2000 by PLATO Learning, Inc.

- *Processing power*. This is determined primarily by the type of processor (e.g., Pentium III), the speed of the computer's data bus (e.g., 300 Megahertz), and the amount of available random access memory (RAM) (e.g., 64 Megabytes).

- Compatibility of required *peripherals*. This includes speed of CD-ROM drive, type of audio card, capabilities of video display, type and speed of printer, etc.

- *Storage space* needed by the software for itself and its learner data (if any). This will be on the learner workstation's hard drive, and may also include space on the network's file server or on the provider's Internet server. Stated in Megabytes (Mb) or Gigabytes (Gb; 1 Gb = 1,000 Mb).

- *Network capacity*. For Local Area Networks (LANs), this includes number of workstations connected, network speed and maximum data transfer rate (measured in megabytes per second). For Internet connections, these factors are important, but it's also important to determine actual connection reliability and data transfer rate (measured in Kb, or kilobytes per second). Insist on real-world measurements during the days of the week and hours of the day when the software will be in use, using the actual connections which you plan to use and the number of simultaneous users you plan.

- Ease of *installation and setup* of the software. Setup must be within the technical capabilities and time availability of you or your technical support people. A major advantage of Internet delivery via a browser is that little or no installation and setup is needed. Software providers may be able to provide installation services for a supplemental fee.

- *Technical support* is critical both during initial installation and for the life of the software. The larger and more complex the software, the more important it is that you protect your investment with technical support. Check for availability and completeness of support information, whether provided by web site, e-mail, telephone, or on-site technician visits. Check for service level agreements and costs, including response time policies. Also, check policies for software upgrades. User training and user groups may also be desirable. A major advantage of Internet delivery is that upgrades usually are automatic.

- *Security*. The software should be reasonably resistant to abusive interference or data theft by "hackers." Many aspects of security are handled at the level of the network and operating system, but individual software products should provide a degree of access security appropriate to its usage context. For example, a simple program on a single CD-ROM which does not store student data may need no security, while a program which delivers tests over a network or the Internet may require substantial security. In networked environments, it's also important for different types of users (and even individuals) to be allowed to do only the things they need to.

Copyright 2000 by PLATO Learning, Inc.

- *Stability and Speed.* The software should be acceptably free of obvious errors, and operation should be stable and acceptably fast during normal use on the minimum platform. Complete reliability is unfortunately beyond the state of the art; any software can be made to "crash" or lose stability if you operate it using hardware or operating system software for which it was not designed, or if you operate it in abnormal ways. But under the manufacturer's specified conditions, you should check to make sure the software operates with acceptable stability and speed. Provision for making backup copies of critical data is a plus.

Table 3 summarizes the criteria for the software subsystem.

Copyright 2000 by PLATO Learning, Inc.

The Software Subsystem Should:

- Be compatible with the available *operating system* & version
- Require no more than the *processing power* and *memory* on the available computers
- Run with the available type and speed of video display & card, audio card, CD-ROM, printer and other peripheral devices
- Use no more than available hard drive *storage space* on the work station/network server for program software & learner data
- Require no more than available *network/Internet capacity* and reliability
- Be *installable* and configurable by personnel available.
- Provide acceptably complete and responsive *answers to technical questions* and software upgrades via Web site, e-mail, hotline, user training, and on-site service calls as appropriate.
- Provide a level of *security* appropriate to the context.
- Be acceptably *error-free, stable and fast* in normal use

Table 3: Criteria for the Software Subsystem

The Instructional Subsystem

Regardless of its type, educational software needs to be designed to fill a particular role in a plan of curriculum and instruction. To determine how well it will fill its role, you need to look at six factors: the software's organization, internal consistency, learner control, flexibility, interactivity, and the way in which it defines the teacher's role. For each of these factors, you can rate how well the software meets your expectations and needs.

Organization. Look at the way information and functions are presented. Check to see if the organization will be logical to your learners, whether it will help your learners to see both the "big picture" and the details, whether any information is well-sequenced from easy to difficult knowledge, and whether the individual components will be appropriately "bite sized" chunks presented at the right pace for your learners. Verbal and visual overviews and summaries often are helpful.

Internal consistency. Compare the goals, objectives, presentation, practice and the assessment (for assessing learner performance). If these components are included, they may be located in the accompanying documentation or in the software itself.

Copyright 2000 by PLATO Learning, Inc.

Check to see if the objectives and assessment correspond in content and level of Bloom's *Taxonomy*. Unfortunately, it's common for software to have very ambitious goals and objectives about "understanding" or "critical thinking," and then to provide interaction and assessment only at a low level of Bloom's taxonomy—or none at all. If the components are vague, inconsistent with each other or missing, there's a good chance the software will be instructionally ineffective and hard to integrate into your curriculum.

Learner Control. Learner control refers to options provided in the software which permit the learner to make decisions regarding what topics to study and which path to follow through out the software (Reeves, 1994). In general, learners like a high degree of learner control, but they may not use it well unless they can monitor their own learning progress. Check to see what kind of, and how much, learner control is provided in the software, and if the degree and kind of learner control is appropriate for the way you intend to use the software.

Flexibility. Some kinds of software are monolithic in structure, and can be used in only one way. Other kinds of software emphasize modular structure and can be used in many different ways—but may require more preparation on your part to use the software effectively. Be sure the software's flexibility will allow you to use it as you wish.

Interactivity. Frequent interactions between the computer and the learner do not guarantee that the software will be instructionally effective. But the use of many *high-quality instructional transactions* (involving presentation, practice and feedback at the right *Taxonomy* level) is a key to effectiveness (Merrill, 1996). Check what level of Bloom's *Taxonomy* the learner must be thinking at in the interactions, and whether the interactions occur around the most important teaching points in the software. Also, look at whether the software provides thought-provoking diagnostic feedback messages in interactions, or if the software goes no further than telling the learner if he or she is right or wrong. Finally, decide if the software provides enough practice for its purpose with your learners.

Teacher's Role. We believe your role in teaching/learning process should always be active; variation in your role depends on what is being taught and which strategy is best for that purpose. Some software provides a fairly complete instructional environment and provides supports for you to be a "guide on the side." Other software assumes that you will provide most of the lesson structure and whatever lesson components are missing from the software. It's important for you to understand what you will need to do to use the software effectively.

Learner's Role. In general, software should place the learner in an active, rather than passive, role. This can be accomplished through frequent interaction and learner control, as well as frequent opportunities for dialog with the system and with peers. Some types of software are designed for personal use by a single learner, while others are designed to support small groups of learners working

Copyright 2000 by PLATO Learning, Inc.

together. There is no "one best role," but it's important to choose software which supports the role(s) you are comfortable with in your classroom.

Table 4 summarizes the criteria for rating the instructional subsystem.

The Instructional Subsystem should have:
• *Organization, chunking and pacing* which is clear and understandable to the learners.
• *Internal Consistency* of instructional components' content and *Taxonomy* level.
• *Learner Control* type and degree which is appropriate for the learners and the way they will use it.
• *Flexibility* and modular structure which will allow you to use the software as you want.
• *Interactivity and practice* which are frequent, of the right *Taxonomy* level, and have feedback on wrong answers which addresses the reason for the error, or explains the principles involved.
• *Teacher's Role* which is clear (described in an instructor guide or help system) and suitable for your intended use.
• *Learner's Role* which is suitable for the instructional model and your classroom.

Table 4. Instructional System criteria

Content/Information Subsystem

Incomplete content is nothing more than sloppy teaching (Laurentiis, 1993). Whether its type is informational or instructional, educational software often fails to present content in complete form and with clear organization. You can judge the content on these criteria:

Definition. Look at the learning outcomes/objectives or table of contents in detail, and then find the corresponding sections in the software. Check to see if the content and its objectives or titles are well defined, and match in subject, level of detail, and (for objectives) level of Bloom's *Taxonomy of the Cognitive Domain.*

Completeness and Accuracy. Content should be correct, current, accurate and complete at the level of detail appropriate for the learners and the purpose, assuming neither too much nor too little prior knowledge. The "big ideas" and central concepts of the topic should be emphasized, and the connections within and between topics and disciplines should be explicit.

Alignment. Look at how well the objectives or topics of the software correspond to the objectives of your curriculum. Be sure to check for correspondence of both topic and level of Bloom's *Taxonomy.* The degree of coverage will help you

Copyright 2000 by PLATO Learning, Inc.

determine where in the curriculum you can use the software and how much benefit you can expect from it. If the software presents the right topics but at too low a *Taxonomy* level (e.g., without opportunity for critical thinking), or with insufficient detail, it will be useful only for supplementary purposes. If the software teaches only a single concept or skill relevant to your curriculum, it will only be useful for the short time your curriculum allocates to those topics. The more complete the alignment, the easier it will be to transfer the software's learning outcomes to the rest of your curriculum.

Examples and Analogies. Look especially at the examples, analogies and imagery used in explanations—both verbal and visual. It's unfortunately common for examples and analogies to be confusing or even off-target relative to the teaching point, or beyond the experience of the learners. It's also unfortunately common to include few, or even no, examples and analogies. Within reason, the more (well-constructed) examples and analogies the software has, the better.

Layout. The content should be laid out in a progression of screens or screen components in a clear and logical fashion using sound typographic and screen layout practices. The layout should help your learners understand the content's logical structure, as well as help them use the software. Note that layout strongly influences readability, and is a significant contributor to reading comprehension even though conventional readability formulas ignore it.

Graphics, Visualization and Multimedia. Software should not be crowded with irrelevant illustrations, whether in graphics, video or audio (sound effects). They can distract learners from the key points rather than adding interest, and may even be unintentionally misleading. Instead, educational software should effectively present content, using whatever level of graphic visualization, video and audio are needed and relevant for clear communication, and which will be appealing to your learners.

Prior Knowledge. All communication assumes that the learner already knows some things. Ask yourself if the software's assumptions about the learner's prior knowledge are appropriate to your learners.

Learner Appropriateness. Look again at the level of detail of information presented, the frame of reference, language, reading level, pacing, and the analogies and examples used for presenting information both verbally and visually. Make sure they are appropriate for your learners. If your learners have special access needs or limited language abilities, make sure there are adequate accessibility provisions for them.

Bias. Make sure the content is free from bias and stereotyping, such as physical, racial, cultural or gender.

Table 5 summarizes the criteria to evaluate the content subsystem of educational software. Obviously, these criteria do not apply to *tools*.

Copyright 2000 by PLATO Learning, Inc.

The Content/Information Subsystem should have
• *Content* which is clearly defined.
• Content which is *complete and accurate* for the purpose and the learner.
• Content which is *aligned* to the curriculum in both scope and *Taxonomy* level.
• Sufficient *examples and analogies* which will be clear to the learners.
• *Layout* will help learners understand the content's logical structure.
• *Reading level* appropriate to the learners.
• *Graphics, visualization and multimedia* if needed and relevant, and which are appealing to your learners
• *Prior knowledge* assumptions which correspond to your learners'
• Frame of reference, language and examples and imagery which are *appropriate for the intended learners.*
• Adequate *accessibility* for your learners.
• Content is *free of bias* or stereotypes

Table 5. Content/Information System

Instructional Management & Assessment Subsystem

The Instructional Management and Assessment subsystem of the Learning System helps you:

- create and save students records (individual as well as group)
- define learning paths and curriculum structures.
- provide appropriate assessments before, during and after instruction.
- make individual decisions about learning prescriptions and study assignments.
- manage access to learning resources.
- compute basic summary statistics you need to monitor and make intervention decisions.
- prepare and print reports for use by you, the learners, administrators and parents.

Whether your learners are on the Web, a local area network (LAN), or shuffling CD-ROMs, the more you use educational software, the more important instructional management becomes. The more you individualize your teaching, the more important automated assessment and instructional management becomes.

Copyright 2000 by PLATO Learning, Inc.

Management is equally important regardless of the type of software you are using. Therefore, when selecting educational software, it's important to consider management features of the software as well as its instructional adequacy (Sloane & et al., 1989). Let's discuss each basic function.

Student Records. If the system maintains student records, then it may be possible for the learners to quit in the middle of a task and resume later where they left off. The records system also may allow you to review the progress of individuals and groups of learners. The more information the system stores about each learner, the better the decisions you may be able to make about each learner. If the information is stored centrally, then learners may be able to use any work station on the network to resume their work, even elsewhere in your building or at home via the Internet. You may want the records shared among administrative and instructional software systems so you don't have to retype anything. You may need a system which allows you to keep your records separate from those of other instructors, classes and groups. However, such power sometimes adds complexity, so there is often a tradeoff between flexibility and ease of use (quality of the "help" system and documentation also is important here). Note also that a system capable of keeping simple records for one class probably will not be capable of tracking hundreds or thousands of learners in a school or district. Therefore, you need to think realistically about your record-keeping needs.

Student records accumulate with use. Therefore, you need a way to write to an archive the records which are no longer needed. This same capability often is associated with facilities for exchanging records with school administrative software, and with making backup copies of student data. Be sure the system you select is capable of managing the number of students you have, both in simultaneous use and in total.

Defining Curriculum Structures and Paths. If you use more than a few educational software products, you will soon want to organize them into groups and sequences which correspond to your curriculum. In larger software curricula, you will want to pick and choose components and sequence them according to your needs. You also will want to intermix software from many different sources, and you may want to intermix on- and offline activities. Thus, you will need to judge the power and flexibility, *vs.* ease of use, with which you can create and use curriculum structures and paths.

Assessment. The more measured learning outcomes are important, the more testing and other forms of assessment are important. There are three basic functions of assessment: diagnostic/placement (before instruction), intermediate progress judgements (during instruction), and summative demonstration of achievement (after instruction, as in a unit test, final exam, or state competency test). A fourth function of assessment is really a kind of practice: you may want to use tests simply for content review and practice in test-taking skills, especially when preparing for a high-stakes test.

Copyright 2000 by PLATO Learning, Inc.

Depending on your needs, all four kinds of assessment capabilities may be important.

Any assessment involves making a measurement and making a judgement based on that measurement. Measurement capabilities of software vary widely. PLATO uses a combination of tests in various formats, together with open-ended activities which produce work products you or the learners can judge. Such combination approaches are often called *portfolio assessment.* You need to determine if the available assessment capabilities will allow you to make the assessment decisions you need to, and whether the tests or other assessments align with your curriculum.

Quality of the tests or other measures themselves also is important. In this review procedure, you judge the quality of measures as part of the *content* and *instruction* subsystems.

Prescription. You need to make sure each learner is assigned the right software component (or Web resource) at the right time. If you used assessment for placement or progress monitoring, you need a system which will make assignments based on the assessment results, or which will help you to do so. You need to determine if the available prescription capabilities will meet your needs, and how flexible, powerful, and easy-to-use the prescription capabilities are.

Access. Even if the learners are in self-guided study, you will want them to be able to select and work through individual learning paths using the options you defined. Thus, you will want a management system which gives you control over what software and web resources the learner can gain access to at any given time. At one extreme, you may want the system to give the learner just one choice about what to do next. At the other extreme, you may want the system to provide a menu of all the potentially relevant activities. In a middle ground, your access management plan may place some limits on what the learner can access, while providing some limited choices to the learner. Whatever your needs, you should choose an instructional management subsystem which can meet them.

Statistics and Reports. The statistics you need, and the summary reports you need to print, should be determined by the information you, your administrators and your parents need. The information you need is determined by the decisions you must make. For example, if you individualize, you may want detailed diagnostic information for individual learners, and the ability to define and track groups of learners or whole classes. You may also want to know time-on-task and other usage information as well as mastery, completion and score information. You also may need a system that can help you identify learners who need special attention, or one that helps you interpret the data. Once you know the kinds of decisions you, your administrators and your parents will be making, you can determine if the available statistics and reporting capabilities will meet your needs, and how flexible, powerful, and easy-to-use these capabilities are.

Copyright 2000 by PLATO Learning, Inc.

Table 6 summarizes the criteria for the instructional management and assessment subsystem.

The Instructional Management & Assessment System Should:

- Record student, subgroup and group records, including identification, progress, and demographic data if needed.
- Perform student record import, export, archive and backup/restore.
- Provide for easy selection and sequencing of available on- and off-line learner activities by individual, subgroup and group, preferably from multiple vendors.
- Support easy definition of curriculum structures and paths.
- Manage and control access to resources by various types and groups of users.
- Have sufficient capacity and performance for the number of users, classes, instructors and total records expected
- Provide for connections on the local computer, local area network or the Internet, depending on your needs.
- Allow learner to exit, save work, and resume work at that point later on.
- Provide adequate methods for assessing student performance for placement, progress monitoring, and summative assessment.
- Make, and/or support making of, prescriptions based on assessment and progress records.
- Provide necessary instructions for proper record keeping, through on-line help system and/or user documentation.
- Provide information to support the decisions made by you, administrators or parents. This may include personal reports, summary tables, charts and graphs describing response patterns and score/grade/mastery obtained, basic statistics of tests, etc.

Table 6. Criteria for the Instructional Management & Assessment System

Additional Criteria for the Five Types of Educational Software

Each of the five types of educational software has its own criteria of quality. When you review a given piece of software, you must first decide which type of software it is, and review it using the four subsystem criteria above (content and

Copyright 2000 by PLATO Learning, Inc.

instruction are especially important for learning). Then, you can complete your review according to the criteria below for that software type.

Drill-and-Practice

Drill and practice is mostly used for knowledge and skill reinforcement. It usually does not intend to teach new knowledge and skills, and rarely goes beyond the Knowledge or Comprehension level ("knowing that") of Bloom's *Taxonomy of the Cognitive Domain*.

Drill and Practice software often combines fact-level presentations (such as memorization of historical events, names, dates and places, vocabulary, or the multiplication tables) with fluency building (such as rapid recall of facts or computation of math problems). Fantasy settings and game structures (alien invasions, detective puzzles, quiz shows, timed races, etc.) often add a "play" element for motivation, but often the "play" is unrelated to the cognitive task being practiced. Most "edutainment" is of this type.

Note that tests and other measures can be examples of drill and practice, if they are used as a means of review and practice (often to prepare for a high-stakes test).

Some software combines drill and practice with information (see below), in the form of fact lists, maps, pictures, and video clips or text explanations (without interaction or feedback beyond simple navigation). PLATO tutorials often are accompanied by application practice, most often at the Application level of Bloom's *Taxonomy*.

Table 7 summarizes the additional quality criteria for drill and practice software.

Copyright 2000 by PLATO Learning, Inc.

Well-designed drill-and-practice software should:

- Provide ample opportunities for practicing a particular skill.
- Provide practice in the desired direction of performance (from cue to response).
- Randomly sequence the elements practiced.
- Use relevant criteria to judge responses (often correctness and sometimes speed of response).
- Provide immediate and appropriate explanatory feedback based on the criteria for a correct answer ("No, that's not right because…").
- Provide additional practice for facts/skills not mastered.
- Provide progressive levels of difficulty, if appropriate to the content and purpose.
- Contain multimedia elements as appropriate to the content.
- Keep learner interest and motivation in the program in a way which supports the intended learning outcome (rather than distracting from it).
- Provide meaningful interaction between user and the content included in the program (not just "click to continue" or interactions relevant only to the game).

Table 7: Additional Criteria for Drill-and-Practice Software

Tutorials

This type of educational software is a computerized form of one-on-one tutoring. PLATO tutorials are mostly used for building well-defined knowledge and skills, including concepts ("knowing what"), principles ("knowing why") and well-defined procedures ("knowing how"). Teaching is usually at the Comprehension and Application levels of Bloom's *Taxonomy of the Cognitive Domain*, though some tutorials reach to the Analysis and Synthesis level. Tutorials can be used to teach facts ("knowing that") at the Knowledge and Comprehension levels of Bloom's *Taxonomy*, but this is rare in PLATO curricula.

Do not confuse tutorial lessons, which have frequent, active practice with feedback, with informational software, which typically includes interactions only to navigate around the content, and otherwise is passive.

Copyright 2000 by PLATO Learning, Inc.

Do not confuse tutorial lessons, which teach concepts, principles and skills, with drill and practice, which only builds proficiency in well-defined procedures already learned elsewhere, or which teach and practice only facts to memorize.

Table 8 summarizes key quality criteria for tutorial software.

Well-designed tutorial software should:

- Teach *well-defined objectives* which accurately describe the content and *Taxonomy* level of what is taught in each lesson.
- Start each lesson with an *orientation/overview*.
- Present information in *small and logically sequenced segments*.
- Size segments so they contain (for adolescents and adults) *up to 5-9 teaching points each*, in *up to 20-30 minutes* of study. For difficult content and for elementary-aged children, fewer teaching points and shorter study times are preferable.
- Provide *guidance* throughout the lesson to both the *knowledge* being learned and the *learning process* itself, through suggestions, symbolic cues, and feedback.
- Include *meaningful questions or interactions* (not just navigation) for each teaching point, keyed to each teaching point and presentation segment.
- Provide appropriate *diagnostic/explanatory feedback* on learner responses to the questions/interactions, especially for learner errors.
- *Model the right answer* if the learner gets "stuck" in an interaction.
- *Jump*, based on the learner's performance and goals, either automatically or by learner choice, to additional topics, examples or more practice in varied formats.

Table 8: Additional Criteria for Tutorial Software

Instructional Simulations and Games

In general, instructional simulations replicate or imitate some aspect of the real world. They place the learner in the role of an actor in the context being simulated. When the learner takes an action, the simulation responds as the real

Copyright 2000 by PLATO Learning, Inc.

system would—whether good or bad. Thus, simulations often allow many "paths" to a solution, and allow more than one outcome. Simulations used for problem solving usually include a scenario which sets the stage for the learner's problem-solving activity and include a clearly defined problem-solving goal.

Simulations are used in situations where real-world experience is unacceptably costly (in resources or time), impossible or risky. For example, learners can do the math to work out a schedule and budget for building a road, and then "build" the road according to their plan, on the computer. They can see the results of their actions in an hour or two, without the cost of a real construction project.

It's not necessarily good for a simulation to be perfectly realistic. For example, a simulation can simplify a process or problem, or exaggerate or compress time or space, in order to make phenomena more easily observable and understandable. Depending on the educational goal and the learners' interests, simulations can use fantasy or authentic contexts (simulations with fantasy contexts are sometimes called *games*; simulations with particularly simplified and flexible representations of reality are sometimes called *microworlds*). However, simulations always respond in a way which imitates how the "real" system would. It's important for the simulation to replicate accurately those portions of reality which the learner will need later to apply what's been learned to other real-world problems. Unfortunately, many arcade-style "edutainment" simulations and games ignore this requirement.

A common problem with simulation is that learners can "play" them mindlessly, like an arcade game. To minimize this problem, instructional simulations can add additional feedback to the learner about the actions he or she takes, the state of the simulation, or the learner's strategy. This kind of instructional feedback is often called *coaching*, and it can help the learner think more deeply about the simulation. It can greatly increase the learning value of a simulation, particularly when teaching problem-solving strategy. Coaching is also an important questioning skill for teachers who use simulations, whether or not the simulation has its own coach.

Table 9 summarizes the criteria for instructional simulations.

Copyright 2000 by PLATO Learning, Inc.

Well-designed instructional simulation software should:

- Provide imaginary experience of a real world or fantasy context which *reproduces those parts of reality needed for transfer* to other problems and contexts.
- Provide a *clear problem scenario* with clear goals.
- Provide all the necessary *rules of the game* and simulation-specific information needed to complete the task, without having to hunt for them.
- Include *all the key information and action steps* used in reality to solve similar problems (simplifying as necessary in early problems to provide *scaffolding*).
- Make *visible phenomena necessary to understanding*, even if they are not visible in reality.
- allow the learner to make the *decisions for each key step* in solving the problem (allowing for scaffolding to skip or simplify steps when appropriate)
- Provide a plausible *range of decision/action options* for each key step, without unrealistic structure (except as needed to provide scaffolding in early experiences).
- Provide *realistic and plausible consequences* for learner responses.
- Provide necessary *tools and information* references needed to solve the problem.
- If teaching of problem solving is a goal, provide *coaching and feedback* on learners' actions which will stimulate reflection on strategy.
- Provide *variations* to allow replay after reflective thought.
- enhance the process of transfer of learning by using *realistic scenarios*, and by *simulating reflection* on basic principles and strategies

Table 9: Additional Criteria for Instructional Simulations and Games

Informational Software

This type of educational software includes on-line databases, most WorldWide Web Sites, and multimedia encyclopedias and references. Informational software is not designed to teach, though it is often useful for learning. While it may have explanations, software of this type typically lacks the instructional features of the other types of software, particularly practice and feedback (you or the learner must

Copyright 2000 by PLATO Learning, Inc.

provide them by other means). Without these features, learning outcomes are unpredictable. Informational software is used mostly as resource tool, primarily for facts and data. Information may be in the form of text or multimedia. It may be specially designed for use by students, or it may be designed for use by general audiences for a range of purposes.

In instructional applications, the main advantage of informational software is that it automates retrieval of factual information. This can greatly increase the efficiency of learner investigations, and provide access to a much broader range of information than is typically available in a classroom, home, or library. This kind of "just in time" availability can even offset the need for some fact learning, and can facilitate active and spontaneous learner exploration of interesting topics. A possible benefit is availability of more classroom time for activities other than information presentation.

But efficiency is not the same as effectiveness. Inadequate or illogical indexing and poor search tools and retrieval strategies can cause the learner to be quickly buried with hundreds or thousands of bits of irrelevant and poorly organized information. The results of the learner's effort can be a search which is *too* efficient, and very ineffective.

Informational software seems to be particularly prone to incorporating multimedia features which add nothing except entertainment (or "eye candy" distraction), in the name of adding interest value. The best use of multimedia is to convey visual and/or auditory information which is important and meaningful to the core message, or to the context of that message. Such uses are equally interesting, but much more effective, than "eye candy."

As with any reference, it's important for the information to be relevant to the curriculum, and complete and accurate enough to serve its educational purpose. It also must be appropriate to the learners' capabilities and frame of reference.

Informational software makes relatively fewer assumptions than the previous types about the purpose of use and the capabilities of the user. However, it does make some, and it's important for those assumptions to suit the purposes and users you anticipate.

Additional criteria for informational software are summarized in Table 10.

Copyright 2000 by PLATO Learning, Inc.

Well-designed informational software should:
• Include content at the right level of completeness and accuracy for the intended use and for the learners, including connections to primary source material as appropriate.
• Encourage critical assessment of information sources.
• have graphics and multimedia features used to aid interpretation, and convey significant information and/or context, if there is a need
• provide a search/exploratory environment which provides efficient *and* effective retrieval
• be organized using a defined knowledge structure which is easily understood by the learner

Table 10: Additional Criteria for Informational Software

Tools

Tools include software which has the purpose of performing some kind of transformation on a defined class of input data, to produce a defined class of output(s). For example, a spreadsheet takes numeric and alphanumeric format data as input to a matrix, performs matrix calculations, and produces a computed spreadsheet as the main output. Graphing software takes the same input and produces a graph or chart of a specified type. Presentation software takes words, media files, and specifications and assembles them into a finished presentation. Communication software connects participants in the on-line learning community, regardless of location.

In addition to assumptions about the tasks for which they will be used, tools also make assumptions about who their users will be and what their goals will be, and it's important for those assumptions to match the users and the purposes you anticipate. This means that tools useful in educational settings must be able to perform tasks which are part of the curriculum, in ways which correspond to the curriculum. They must be easily mastered by the learners (and teachers!), and they must use language and assume knowledge and have a frame of reference which corresponds to that of the learners. Thus, for example, a word processor might support common writing functions, and include a spelling and grammar/syntax checker, but it would be especially useful if it had supports for the entire pre-writing process built in, and supports group/peer reviews of drafts as is commonly done in writing classes.

Copyright 2000 by PLATO Learning, Inc.

Tools generally do not make judgements or provide feedback about inputs and outputs the way drill and practice, tutorials and simulations do. In educational uses, these judgements and feedback must be provided by you or peer learners.

The lack of judgements and relatively flexible structure of tools allow them to be used in relatively realistic ways. This is important when the educational goal is transfer, exploration, or open-ended investigation or creation. However, it also may be useful to have a way to add scaffolding to the task in order to simplify it early in the learning process. It may also be useful to provide advanced help features which model tasks and goals.

Table 11 summarizes the additional criteria for tools.

Tool Software Should:
• Support whole, defined tasks as they are defined and described in the curriculum.
• Make appropriate assumptions about the learner's goals, skills and prior knowledge.
• Provide support for instructionally useful features such as storing work, tracking revisions, teacher and peer review/feedback.
• Scaffold and model tasks as appropriate
• Link together to share information among tools and among learners as appropriate.

Table 11: Additional Criteria for Tool Software

Which Type of Software Should I Use?

The range of software types available for educational use can sometimes be confusing. It's unfortunately common for instructors to find just one or two types which seem to work well, and stick to them without further experimentation. This is especially likely to happen if the first examples you encounter of a new software type happen to be of mediocre quality. It's also unfortunately common for "experts" to advocate one type of software for exclusive use in classrooms, without regard to the advantages and limitations of each type.

We argue for a more balanced view. Each of the software types described above represents a different cluster of feature. Assuming good design, each type is likely to be effective for different purposes. While classroom instructional practices differ, we believe there is probably a place for nearly every type of software in nearly every type of classroom – though the role and overall importance of the software may change from one classroom to another. In short, there is no "one

Copyright 2000 by PLATO Learning, Inc.

best way" to use educational software, and there is no "one best type" of software to use.

To illustrate the point, let's examine two hypothetical secondary language arts classrooms, taught by Miss Jones and Mr. Smith.

Miss Jones' class is based on constructivist principles. She likes to have learners explore and make their own meanings by using the rich learning environment she creates. In Miss Jones' class, each type of software has a role to play.

- She often starts a unit with a *simulation* or *game* based on an issue related to the theme of the unit. Learners work collaboratively in small groups, debating the goals and strategies for the game.

- She then helps her students formulate issues or problems to study, which are related to the situations first encountered in the simulations or games. In the course of learning about their problems or issues, her students work independently and in small groups using *informational* software and the Internet.

- By no coincidence, the investigations often time require a deeper mastery of concepts and skills in various content areas, as well as in the skills of reading and comprehension. As learners come upon the need to deepen their understanding of concepts and their mastery of skills, Miss Jones can assign *tutorial and practice* software to individuals based on their needs and goals.

- Each investigation concludes with a work product such as a written report, slide presentation, or web site produced by the collaborative teams, using a variety of *tools*.

- Even though the class is highly individualized, Miss Jones uses the *management* and *assessment* capabilities of her software to track the progress of each individual learner against the local curriculum standards, and to quickly identify those learners who may need intervention. Without such automated assistance, the mountain of papers to grade and individual decisions to make each day would quickly become overwhelming.

Mr. Smith's class has one overarching goal: prepare the learners to read and write well enough so they can pass the state competency test looming at the end of the year, and go on to high school. In Mr. Smith's class, each type of software has a role to play.

- At the beginning of the semester, all of Mr. Smith's students took an automated *placement test* in basic skills of reading and writing. The computers then automatically placed each learner at the right points in the

Copyright 2000 by PLATO Learning, Inc.

curricula for reading and writing which Mr. Smith had designed based on his analysis of the state standards and the software available.

- Each student works on his or her own personal sequence of *tutorial* and *practice* software, mastering the skills of reading and writing.

- Mr. Smith requires quite a bit of reading and writing time in class and out. With the help of his library-media specialist, he has gathered together a variety of Web sites and local CD-ROM *informational* resources which are at the right levels for his learners and on high-interest topics. In addition, he always has his learners working offline on various worksheets and various books.

- As the learners work, Mr. Smith encourages them to pair up and work together. He's found that learners of different abilities but common goals often work together most successfully. He encourages each pair of learners to use various *tools* to engage in information organizing and argumentation, through construction of tables, mind maps, etc. – all of which ultimately get turned into essays.

- Mr. Smith uses *simulations and games* as integrative activities for summation and integration. Once his learners have mastered the underlying skills, Mr. Smith can use the appropriate *simulation or game* as a way to get the learners to review and reinforce what they have learned, while developing confidence in their newly-acquired skills.

- Mr. Smith can individualize his classroom, but he still knows where every learner is and what they are finding difficult, with the help of the *management an assessment* system.

In each of these examples, the teachers are in the "guide on the side" role, using the computers to do much of the "heavy lifting" of skill acquisition. For further discussion of this issue, see *PLATO Technical Paper #6.*

Copyright 2000 by PLATO Learning, Inc.

Part 4

A Checklist for Reviewing Educational Software

The checklist which begins on the next page outlines a process involving 5 global judgements you should make about any educational software product. Look at the beginning, the middle, and the end parts of each significant section of the software, then perform these reviews:

For all software types:

1. Review the software system
2. Review the management/assessment system

For all software types except tools:

3. Review the content system

For all software types except tools and information:

4. Review the instructional system

If the results of any of the above reviews cause you to reject the product, STOP.

If the candidate product has not been ruled out at this point, complete the review using the additional criteria for the software types included in the product. Select from supplementary checklists for:

- ***Drill & Practice***
- ***Tutorial***
- ***Simulation***
- ***Information***
- ***Tool***

Copyright 2000 by PLATO Learning, Inc.

PLATO® *Software Review Checklist*

Title	**Publisher**
Curriculum ❑ **Math** ❑ **Reading** ❑ **Writing** ❑ **Science** ❑ **Other**	**Labeled Principal Grade Level (for series)** ❑ **Pre-K** ❑ **K-3** ❑ **4-6** ❑ **6-9** ❑ **9-12** ❑ **12-14 & Adult**
Labeled or Estimated Hours of Activity by Principal Learners _____ **Hours for this part;** _____ **Hours for entire series**	
Reviewer	**Review Date**

Product Includes These Software Types:
❑ Drill & Practice ❑ Tutorial ❑ Simulation ❑ Information ❑ Tool

Sample beginning, middle, end of a part which is representative of the series.

Software Subsystem	**Does not meet our needs** 1	**Useable; meets some of our needs** 2	**Meets most of our needs** 3	**Meets all of our needs** 4	**Exceeds our needs/ room for growth** 5

- ❑ Be compatible with the available *operating system* & version
- ❑ Require no more than the *processing power* and *memory* on the available computers
- ❑ Run with the available type and speed of video display & card, audio card, CD-ROM, printer and other peripheral devices
- ❑ Use no more than available hard drive *storage space* on the work station/network server for program software & learner data
- ❑ Require no more than available *network/Internet capacity* and reliability
- ❑ Be *installable* and configurable by personnel available.
- ❑ Provide acceptably complete and responsive *answers to technical questions* and software upgrades via Web site, e-mail, hotline, user training, and on-site service calls as appropriate.
- ❑ Provide a level of *security* appropriate to the context.
- ❑ Be acceptably *error-free, stable and fast* in normal use

Copyright 2000 by PLATO Learning, Inc.

Management/Assessment Subsystem	**Does not meet our needs** **1**	**Useable; meets some of our needs** **2**	**Meets most of our needs** **3**	**Meets all of our needs** **4**	**Exceeds our needs/ room for growth** **5**
❑ Record student, subgroup and group records, including identification, progress, and demographic data if needed. ❑ Perform student record import, export, archive and backup/restore. ❑ Provide for easy selection and sequencing of available on- and off-line learner activities by individual, subgroup and group, preferably from multiple vendors. ❑ Support easy definition of curriculum structures and paths. ❑ Manage and control access to resources by various types and groups of users. ❑ Have sufficient capacity and performance for the number of users, classes, instructors and total records expected ❑ Provide for connections on the local computer, local area network or the Internet, depending on your needs. ❑ Allow learner to exit, save work, and resume work at that point later on. ❑ Provide adequate methods for assessing student performance for placement, progress monitoring, and summative assessment. ❑ Make, and/or support making of, prescriptions based on assessment and progress records. ❑ Provide necessary instructions for proper record keeping, through on-line help system and/or user documentation. ❑ Provide information to support the decisions made by you, administrators or parents. This may include personal reports, summary tables, charts and graphs describing response patterns and score/grade/mastery obtained, basic statistics of tests, etc.					

Content/Information Subsystem	**Does not meet our needs** **1**	**Useable; meets some of our needs** **2**	**Meets most of our needs** **3**	**Meets all of our needs** **4**	**Exceeds our needs/ room for growth** **5**
❑ *Content* which is clearly defined (e.g., through use of objectives). ❑ Content which is *complete and accurate* for the purpose and the learner. ❑ Content which is *aligned* to the curriculum in both scope and *Taxonomy* level. ❑ Sufficient *examples and analogies* which will be clear to the learners. ❑ *Layout* will help learners understand the content's logical structure. ❑ *Reading level* appropriate to the learners. ❑ *Graphics, visualization and multimedia* if needed and relevant, and which are appealing to your learners ❑ *Prior knowledge* assumptions which correspond to your learners' ❑ Frame of reference, language and examples and imagery which are *appropriate for the intended learners.* ❑ Adequate *accessibility* for your learners. ❑ Content is *free of bias* or stereotypes ❑ *Teacher's Role* is clear (described in instructor guide or help system) and suitable for the way software is intended to be used.					

Copyright 2000 by PLATO Learning, Inc.

Instructional Subsystem	**Does not meet our needs** **1**	**Useable; meets some of our needs** **2**	**Meets most of our needs** **3**	**Meets all of our needs** **4**	**Exceeds our needs/ room for growth** **5**
❑ *Organization, chunking and pacing* which is clear and understandable to the learners. ❑ *Internal Consistency* of instructional components' content and *Taxonomy* level. ❑ *Learner Control* type and degree which is appropriate for the learners and the way they will use it. ❑ *Flexibility* and modular structure which will allow you to use the software as you want. ❑ *Interactivity and practice* which are frequent, of the right *Taxonomy* level, and have feedback on wrong answers which addresses the reason for the error, or explains the principles involved. ❑ *Teacher's Role* which is clear and suitable for your intended use. ❑ *Learner's Role* which is suitable for the instructional model and your classroom.					

Subtotal for Subsystem Review_____

Software Type Ratings

Complete all that apply

❑ Software includes drill-and-practice or game components.

Does not meet our needs **1**	**Useable; meets some of our needs** **2**	**Meets most of our needs** **3**	**Meets all of our needs** **4**	**Exceeds our needs/ room for growth** **5**
❑ provide ample opportunities for practicing a particular skill. ❑ provide practice in the desired direction of performance (from cue to response). ❑ randomly sequence the elements practiced. ❑ use relevant criteria to judge responses (often correctness and sometimes speed of response). ❑ provide immediate and appropriate explanatory feedback based on the criteria for a correct answer ("No, that's not right because…"). ❑ provide additional practice for facts/skills not mastered. ❑ provide progressive levels of difficulty, if appropriate to the content and purpose. ❑ contain multimedia elements as appropriate to the content. ❑ keep learner interest and motivation in the program in a way which supports the intended learning outcome (rather than distracting from it). ❑ provide meaningful interaction between user and the content included in the program (not just "click to continue" or interactions relevant only to the game).				

Copyright 2000 by PLATO Learning, Inc.

❑ Software includes tutorial instruction components.

Does not meet our needs 1	Useable; meets some of our needs 2	Meets most of our needs 3	Meets all of our needs 4	Exceeds our needs/ room for growth 5

- ❑ teach *well-defined objectives* which accurately describe the content and *Taxonomy* level of what is taught in each lesson.
- ❑ start each lesson with an *orientation/overview*.
- ❑ present information in *small and logically sequenced segments*.
- ❑ size segments so they contain (for adolescents and adults) *up to 5-9 teaching points each*, in *up to 20-30 minutes* of study. For difficult content and for elementary-aged children, fewer teaching points and shorter study times are preferable.
- ❑ provide *guidance* throughout the lesson to both the *knowledge* being learned and the *learning process* itself, through suggestions, symbolic cues, and feedback.
- ❑ include *meaningful questions or interactions* (not just navigation) for each teaching point, keyed to each teaching point and presentation segment.
- ❑ provide appropriate *diagnostic/explanatory feedback* on learner responses to the questions/interactions, especially for learner errors.
- ❑ *model the right answer* if the learner gets "stuck" in an interaction.
- ❑ *jump*, based on the learner's performance and goals, either automatically or by learner choice, to additional topics, examples or more practice of varied types.

Copyright 2000 by PLATO Learning, Inc.

❑ Software includes simulation components.

Does not meet our needs 1	Useable; meets some of our needs 2	Meets most of our needs 3	Meets all of our needs 4	Exceeds our needs/ room for growth 5

- ❑ provide imaginary experience of a real world or fantasy context which *reproduces those parts of reality needed for transfer* to other problems and contexts.
- ❑ provide a *clear problem scenario* with clear goals.
- ❑ provide all the necessary *rules of the game* and simulation-specific information needed to complete the task, without having to hunt for them.
- ❑ include *all the key information and action steps* used in reality to solve similar problems (simplifying as necessary in early problems to provide *scaffolding*).
- ❑ Make *visible phenomena necessary to understanding*, even if they are not visible in reality.
- ❑ allow the learner to make the *decisions for each key step* in solving the problem (allowing for scaffolding to skip or simplify steps when appropriate)
- ❑ provide a plausible *range of decision/action options* for each key step, without unrealistic structure (except as needed to provide scaffolding in early experiences).
- ❑ provide *realistic and plausible consequences* for learner responses.
- ❑ provide necessary *tools and information* references needed to solve the problem.
- ❑ if teaching of problem-solving is a goal, provide *coaching and feedback* on learners' actions which will stimulate reflection on strategy.
- ❑ provide *variations* to allow replay after reflective thought.
- ❑ enhance the process of transfer of learning by using *realistic scenarios*, and by *simulating reflection* on basic principles and strategies

❑ Software includes information components.

Does not meet our needs 1	Useable; meets some of our needs 2	Meets most of our needs 3	Meets all of our needs 4	Exceeds our needs/ room for growth 5

- ❑ Include content at the right level of completeness and accuracy for the intended use and for the learners, including connections to primary source material as appropriate.
- ❑ Encourage critical assessment of information sources.
- ❑ have graphics and multimedia features used to aid interpretation, and convey significant information and/or context, if there is a need
- ❑ provide a search/exploratory environment which provides efficient *and* effective retrieval
- ❑ be organized using a defined knowledge structure which is easily understood by the learner

Copyright 2000 by PLATO Learning, Inc.

❑ Software includes tool components.

Does not meet our needs 1	Useable; meets some of our needs 2	Meets most of our needs 3	Meets all of our needs 4	Exceeds our needs/ room for growth 5

- ❑ Support whole, defined tasks as they are defined and described in the curriculum.
- ❑ Make appropriate assumptions about the learner's goals, skills and prior knowledge.
- ❑ Provide support for instructionally useful features such as storing work, tracking revisions, teacher and peer review/feedback.
- ❑ Scaffold and model tasks as appropriate
- ❑ Link together to share information among tools and among learners as appropriate.

Total score (compare only against software of comparable types and scope)

_____

Comments:

Copyright 2000 by PLATO Learning, Inc.

References

De Laurentiis, E.C.(1993). How to recognize excellent educational software. New York: Lifelong Software, Inc.

Doll, Carol A.(1987). Evaluating educational software. Chicago: American Library Association.

Flemming, Malcom lee(1993). Instructional message design: principles from behavioral and cognitive sciences. New Jersey: Educational Technology Publications.

Foshay, Rob(1993). Presentation on PLATO software. Schaumburg: TRO Learning, Inc.

Gagne, Robert M.(1985). The conditions of learning and theory of instruction. Florida: Robert Woodbury.

Gill, Barbara J., & et al.(1992). A new model for evaluating instructional software. Educational Technology, 32(3), 39-48.

Graells, Pere Marques I.(1993). Designing educational software for pupils in the 12-16 age group: what to produce and how? Educational Media International, 30(3), 138-142.

Hakkinen, Paivi(1996). Software designers and teachers as evaluators of computer-based learning environments. Machine-Mediated Learning, 5(2), 135-148.

Hannafin, Michael J., & Peck, Kyle L.(1988). The design, development, and evaluation of instructional software. New York: MacMillan Publishing Company.

Merrill, M. David, Jones, Mark K. & Li, Zhongmin(1992). Instructional transaction theory: classes of transactions. Educational Technology, 32(3), 12-25.

Merrill, M. David & ID2 team(1996). Instructional transaction theory: instructional design based on knowledge objects. Educational Technology, 36(3), 30-37.

Price, Robert V.(1991). Computer-aided instruction: a guide for authors. California: Brooks/Cole Publishing Company.

Reeves, Thomas C.(1991). Ten commandments for the evaluation of interactive multimedia in higher education. Journal of Computing in Higher education, 2(2), 84-113.

Reeves, Thomas C.(1992). Evaluating interactive multimedia. Educational Technology, 32(5), 47-53.

Reeves, Thomas C.(1994). Evaluating what really matters in computer-based education. In Wild, M, & Kirkpatrick, D.(Eds.), Computer education: new perspectives.(219-246) Perth, Australia: MASTEC, Edith Cowan University Press

Copyright 2000 by PLATO Learning, Inc.

Reigeluth, Charles M.(1983). Instructional design theories and models. New Jersey: Lawrence Erlbaum Associate, Inc.

Richards, Thomas C., & Fukuzawa, Jeannette(1989). A checklist for evaluation of courseware authoring systems. Educational Technology, 24-29.

Rushinek, Sara F.(1986). Computer-assisted instruction: performance standards for the design of interactive feedback and control. AEDS, 19(2-3),

Sloane, Howard N., & et al.(1989). Evaluating educational software: a guide for teachers. New Jersey: Prentice Hall, Inc.

Shneiderman, Ben(1993). Designing the user interface: strategies for effective human-computer interaction. New York: Addison-Wesley Publishing Company.

Squires, David, & McDougall, Anne(1994). Choosing and using educational software: a teacher's guide. Washington: The Falmer Press.

Copyright 2000 by PLATO Learning, Inc.